Remarks

Claims 1-14 are currently pending in this application. Claims 1-13 are original. Claim 14 is new.

Claims 1, 2, 4-6 and 9-13 are rejected under 35 U.S.C. 102(b) as being anticipated by Gostanian et al. (US Pat. 5,781,910).

Regarding Claim 1.

Claim 1 recites:

1. A method for executing two or more computational operations upon elements of a data structure, the method comprising the steps of:
(a) determining if any of the two or more computational operations to be executed are operable upon a same element;
(b) determining if any of the two or more computational operations determined to be operable upon the same element are in kind operations;
(c) determining if any of the two or more computational operations determined to be operable upon the same element and to be in kind operations are addition or assignment operations; and
(d) executing the two or more computational operations determined to be operable upon the same element, to be in kind operations, and to be addition operations.

In rejecting Claim 1 the Examiner states:

Gostanian discloses a method in which it is determined that two or more operations are operable on the same element and that the operations are in kind operations (column 10, line 15-column 11, line 10 of Gostanian). It is then determined that the operations are addition operations, thus commutative (column 10, line 15-column 11, line 10 of Gostanian).

The Applicant has reviewed the cited art and is unable to identify any teaching that includes "*determining if any of the two or more computational operations to be executed are operable upon a same element*" as suggested by the Examiner.

For example, there does not appear to be any teaching within Gostanian that includes "*determining if ... computational operations to be executed are operable upon* *the same element*," (emphasis added) as recited in Claim 1. In Gostanian, a transaction is classified as commutative or non-commutative responsive to other transactions not yet requested by a client and, thus, before the elements on which the other transactions will operate are known. For example, Gostanian teaches "[t]ransactions may be commutative or non-commutative depending on the type of transaction and on the other transactions that may be requested by the application clients," (Col. 10 lines 34-37, emphasis added). Further, at Col. 10 lines 54-56, Gostanian teaches that merely "supporting" a second transaction is sufficient to make a commutative transaction non-commutative. Again, this support of a second transaction and resulting reclassification appear to occur before a request for the second transaction is received from a client. Further, because the transactions described in Gostanian are future transactions, the element on which the second transaction will operate is not identifiable until the client's request is received. Thus, in Gostanian, a transaction is classified as commutative or non-commutative before it is possible to determine if it is operating on the same element as other transactions considered in the classification. The classification, therefore, cannot be based on whether "*two or more computational operations to be executed are operable upon a same element*," as recited in Claim 1. In contrast with Gostanian, the Claim 1 step of "*determining if ... operations to be executed are operable upon a same element*," requires knowledge of the "*same element*" at the time the determination is made. Because Gostanian does not teach using this knowledge at the time a transaction is classified as commutative or non-commutative, Gostanian does not teach "*determining if any of the two or more computational operations to be executed are operable upon a same element*," as recited in Claim 1.

The Applicant respectfully points out that as recited in the preamble of Claim 1, the element in the context of Claim 1 is an element "*of a data structure.*" Thus, an entire database as taught in Gostanian would not properly be considered "*a same element*" "*of a data structure*" as recited in Claim 1.

Further, while Gostanian does teach the execution of more than one transaction on the same database, there does not appear to be any teaching that includes a determination regarding "*two or more computational operations to be executed,*" (emphasis added) as recited in Claim 1. Rather, the approach taken by Gostanian appears to be to consider a transaction and all other transactions supported by a system. These considerations are made without regard to which transactions are actually to be executed and which data elements they may be applied to. As pointed out above, at Col. 10 lines 34-37, Gostanian states "[t]ransactions may be commutative or non-commutative depending on the type of transaction and on the other transactions that may be requested by the application clients," (emphasis added). Thus, in Gostanian, the type of transaction and the set of all other transactions that may be requested are used to determine if a transaction is commutative. Thus, Gostanian teaches that whether a transaction is commutative, or not, is not dependant on any consideration of which transactions are actually "*to be executed.*" This is further supported by Col. 10 lines 51-54 of Gostanian which state "[a] transaction is non-commutative if the final state of the database server is dependent on [t]he order in which the transaction is executed relative to other transactions that might be requested by an application client," (emphasis added). The term "might" indicates that the other transactions need not be "*operations to be executed.*" In contrast with the teachings of Gostanian, Claim 1 includes a determination based on "*operations to be executed.*"

Therefore, it is the position of the Applicant that Gostanian does not teach any determination regarding "*any of the two or more computational operations to be executed are operable on the same element*," as recited in Claim 1.

Further, in rejecting Claim 1, the Examiner states that Gostanian teaches "[i]t is then determined that the operations are addition operations, thus commutative (column 10, line 15-column 11, line 10 of Gostanian)," and suggests that this teaches "*determining if any of the two or more computational operations determined to be operable upon the same element are in kind operations*" as recited in Claim 1. The Applicant is unable to find any such teaching in Gostanian. At Col. 10 lines 41-44 Gostanian provides an example wherein all transactions within a system are defined as including addition operations. However, in a system defined in this manner all transactions are the same by definition and there would be no reason to determine "*if any of the two or more computational operations determined to be operable upon the same element are in kind operations*," as recited in Claim 1. In the example of Gostanian, the conditional "*if*" would always be true. The Applicant, therefore respectfully requests that the Examiner clarify how the cited text teaches "*(b) determining if any of the two or more computational operations determined to be operable upon the same element are in kind operations*" or allow Claim 1.

Further, the Applicant is unable to identify any teaching within Gostanian of determination of "*in kind operations*." It is, therefore, the Applicant's position that Gostanian does not teach "*(b) determining if any of the two or more computational operations determined to be operable upon the same element are in kind operations*" as recited in Claim 1. The determination of "*in kind operations*" involves making a comparison between the "*two or more computational operations*" in order to determine if they are of the same kind (application as filed, pg 13, lines 5-16). Therefore, the Applicant requests that the Examiner point out teaching within Gostanian that includes comparison of one operation with another for the purpose of determining whether or not they are "*in kind operations*," or allow Claim 1.

As pointed out above, Gostanian teaches that whether or not a transaction is commutative depends on the type of the transaction and other possible transactions, (Col. 10 lines 34-37). Even if, for the sake of argument, one were to conclude that this implied a comparison between a first transaction and all other possible transactions, there does not appear to be any indication that this comparison includes "*determining if any of the two or more computational elements ... are in kind operations*" as recited in Claim 1.

Gostanian provides two examples of determining whether a transaction is commutative. The first, at Col. 10 lines 41-44, includes a system in which there is only one transaction (addition) allowed by the system. As discussed above, in this system there would be no reason to make any determination that any two specific transactions to be executed "are in kind operations" because all operations are of the same kind. Thus, in this example of Gostanian, the step of "*determining if any of the two or more computational elements ... are in kind operations*," as recited in Claim 1, would be meaningless.

The second example, at Col. 10 lines 51-67 of Gostanian, includes an additional transaction type that makes the transaction of the first example non-commutative. The second transaction and the first transaction happen not to be "*in kind*" transactions. However, the Applicant is unable to identify any teaching within Gostanian that the fact that the transactions are not "*in kind*" is the basis for their classification. Rather, it appears that the classification is the result of trying examples and noting that the result is dependent on transaction order. It is, therefore, the position of the Applicant that the second example also does not teach "*determining if any of the two or more computational elements ... are in kind operations*" as recited in Claim 1. The Applicant, therefore, requests that the Examiner more specifically point out this teaching in the cited art or allow Claim 1.

For at least these reasons, the Applicant believes that Claim 1 is allowable.

Regarding Claim 2.

Claim 2 recites:

2. The method of claim 1 further comprising the steps of:
(e) determining, of the two or more computational operations determined to be operable upon the same element, to be in kind operations, and to be assignment operations, if a same value is to be assigned to the same element; and
(f) executing the two or more computational operations determined to be operable upon the same element, to be in kind operations, to be assignment operations, and to assign the same value to the same element.

In rejecting Claim 2, the Examiner states:

Gostanian discloses a method in which it is determined that the result of two operations is listed as commutative and thus are allowed to execute if it is determined that the result would be identical if the operations were run in any order, which would include two identical assignment statements. (column 10, line 15-column 11, line 10...).

As pointed out above, Gostanian appears to determine whether a transaction is commutative (e.g., whether the order of execution will matter) merely by considering the operation's type and what other transactions are supported. The Applicant is unable to find any further teaching within Gostanian regarding how to determine if an operation is commutative or non-commutative. In an approach such as that of Gostanian, an assignment operation would normally be considered non-commutative based merely on the operation's type because assignments are often non-commutative. (A<=2, A<=3 yields a different result than A<=3, A<=2.) Gostanian does not teach how a commutative assignment operation would be differentiated from a non-commutative assignment operation. Therefore, contrary to the Examiner's suggestion, it would be impossible for Gostanian to identify that an assignment operation is commutative. As a consequence, Gostanian cannot teach "*(e) determining, of the two or more computational operations determined to be operable upon the same element, to be in kind operations, and to be assignment operations, if a same value is to be assigned to the same element*," as recited in Claim 2.

For at least these reasons, and those discussed above with respect to Claim 1, the Applicant believes that Claim 2 is allowable.

Regarding Claims 4-6, 9 and 11.

It is the Applicant's position that Claims 4-6, 9 and 11 are allowable for at least the reasons discussed above with respect to Claim 1.

Regarding Claim 10.

It is the Applicant's position that Claim 10 is allowable for at least the reasons discussed above with respect to Claims 1 and 2.

Regarding Claim 12.

Claim 12 recites:

12. A method for executing two computational operations upon elements of a data structure, the method comprising the steps of:
executing the two computational operations if
either computational operation does not violate a limit, and
both computational operations do not operate upon a same element;
executing the two computational operations if

either computational operation does not violate the limit,
both computational operations operate upon the same element, and
both computational operations are addition operations; and
executing the computational operations if
either computational operation does not violate the limit,
both computational operations operate upon the same element, and
both computational operations are assignment operations that assign a same value to the same element.

In rejecting Claim 12 the Examiner states "Gostanian discloses a method in which computation operations will be executed based on no direct limits (thus no limit is violated)…" The Applicant traverses this statement. The fact that Gostanian does not teach use of limits, does not imply teaching of "*executing the two computational operations **if** either computational operation does not violate a limit*," as recited in Claim 12. It is the position of the Applicant that the language of Claim 12 requires the existence of a limit for the conditional "*if*" to be determined. Further, there must be a limit in order for there to be a limit that is not violated. For example, the language of Claim 12 "*does not violate a limit*" differs from the language "no limit is violated," suggested by the Examiner in the quote above. In the first case (the Claim 12 language) there is a limit that is not violated, while in the second case (the Examiner's language) there may either be a limit that is not violated or no limit at all. It is the Applicant's position that the while the Examiner's arguments could conceivably be applicable in the second case – where there is no limit, the Examiner's arguments are not applicable to the language of Claim 12 which requires the existence of a limit that is not violated. Therefore, because Gostanian does not teach the use of limits in determining if an operation is commutative or non-commutative, Gostanian does not teach "*executing the two computational operations if either computational operation does not violate a limit*," as recited in Claim 12.

Further, the Applicant respectfully points out that the teachings of Gostanian are inconsistent with use of limits in determining if an operation is commutative. Specifically, at Col. 10 lines 41-43, Gostanian teaches that "if the only transaction allowed by the system is a transaction that adds some amount to a deposit account, then these transactions are commutative." The Applicant understands this to indicate that Gostanian makes an assumption that if all transactions are addition transactions, then all the transactions are to be considered commutative on this basis alone. As pointed out on page 14 lines 12-15 of the application as filed, the assumption that addition is always commutative is not necessarily true when limits are considered. Thus, if limits were included in the system of Gostanian then the above assumption taught by Gostanian would fail. Therefore, the system of Gostanian is inconsistent with the use of limits to determine if a computational operation should be executed as recited in Claim 12.

Further, it is the Applicant's position that the cited art does not teach "*executing the two computational operations if ... both computational operations do not operate upon a same element*," as recited in Claim 12. In rejecting Claim 12, the Examiner States "[i]n one case, the operations will be executed if they are not operational on the same elements (column 10, line 15-column 11, line 10 of Gostanian)." The Applicant is unable to find any such teaching within the cited text. Specifically, there does not appear to be any teaching that commutativity of an operation is determined responsive to a conditional determination as to whether another operation operates on the same element. This point was discussed above with respect to Claim 1. The Applicant therefore, respectfully requests that the Examiner more particularly point out this teaching within the Gostanian or allow Claim 12.

Furthermore, it is the Applicant's position that the cited art does not teach "*executing the computational operations if either computational operation does not violate the limit, both computational operations operate upon the same element, and both computational operations are assignment operations that assign a same value to the same element*," as recited in Claim 12. In rejecting Claim 12, the Examiner further states:

> In yet another case they will execute if they operate on the same element and they are assigning the same value to the element, based on the fact that Gostanian discloses a method in which it is determined that the result of two operations is listed as commutative and thus are allowed to execute if it is determined that the result would be identical if the operations were run in any order, which would include two identical assignment statements. (column 10, line 15-column 11, line 10 of Gostanian).

The Applicant is unable to find any such teaching within the cited text. Specifically, while Claim 12 recites a <u>conditional</u> limitation using the work "*if*," there does not appear to be any conditional aspect taught in <u>Gostanian</u> wherein execution of an operation is dependant on a condition that either of two or more operations violate a limit, that both operations operate on the same element, *or* that both operations include assigning a same value.

Further, while <u>Gostanian</u> defines an operation as being commutative when the result of the operation is independent on operation order, this definition does not teach how to identify a commutative operation using the method of Claim 12, or use of the conditional aspects of Claim 12 to identify when two operations may be executed regardless of order. The Applicant, therefore, respectfully requests that the Examiner more particularly point out these teachings within the <u>Gostanian</u> or allow Claim 12.

Regarding Claim 13.

The Applicant believes that Claim 13 is allowable for the same reasons discussed above with respect to Claim 12.

Claims 3, 7 and 8 are rejected under 35 U.S.C. 103(a) as being unpatentable over <u>Gostanian</u>.

Regarding Claim 3.

Claim 3 recites:

3. The method of claim 2 further comprising the step of:
determining if any of the two or more computational operations determined to be operable upon the same element and to be in kind operations violate a limit, then not performing steps (d) or (f).

In rejecting Claim 3 the Examiner states "Gostanian discloses that in the example of a deposit account it would be imperative to process some addition operations (i.e. deposit and withdrawal) in order to keep a deposit account from falling to a negative value when it should not (column 1, lines 40-63 of Gostanian)." The Examiner then makes inferences based on this supposed teaching. Specifically, the Examiner suggests that there is an implication that execution of operations are responsive to violation of the limit.

The Applicant is unable to find any teaching, either express or implied, within the cited text regarding limits, much less deposit accounts with negative values, as suggested by the Examiner. Rather, the cited text concerns transferring funds from one account to another and the requirement that the withdrawal from a *first* account and the deposit into a *second* account must be performed as an atomic process. The Applicant notes that the first account and the second account would be associated with different elements within a data structure. There is no discussion in the cited text of the use of limits to determine if an operation should be executed. As such, there does not appear to be any basis for the Examiner's suggested implication.

In sharp contrast with the cited art and the suggestions made by the Examiner, Claim 3 includes "*determining if any of the two or more computational operations ... violate a limit, then not performing steps (d) or (f).*" Thus, the possibility of violating a limit is examined with regard to at least two different operations and the execution of further steps are responsive to these at least two determinations. The Applicant is unable to identify any such teaching in Gostanian.

Further, in Claim 3, the "two or more computational operations" are "determined to be operable upon the same element" according to independent Claim 1 from which Claim 3 depends. As discussed above, the text cited by the Examiner concerns operations on different elements, representing a first account and a second account.

The Applicant, therefore, requests that the Examiner specifically point out teaching within the cited art that includes making a determination regarding limit violation for "*two or more computational operations*" "*determined to be operable upon the same element*" and performing steps responsive to these determinations, as well as the other limitations of Claim 3, or allow Claim 3,

The Applicant further believes that Claim 3 is allowable for at least the reasons discussed above with respect to Claims 1 and 2.

Regarding Claim 7.

Claim 7 recites:

7. A method for categorizing two or more computational operations executable upon elements of a data structure, the method comprising the steps of:

determining if any of the two or more computational operations violate a limit; and
categorizing the two or more computational operations determined to violate the limit as not commutative.

In rejecting Claim 7 the Examiner again refers to Col. 1 lines 40-63 of <u>Gostanian</u> and suggests that this text teaches use of a limit and that this teaching implies that execution of operations are responsive to violation of the limit. As discussed above, the Applicant is unable to find the express teachings that are suggested by the Examiner, much less a basis for the implications that are suggested to arise from these express teachings. The cited text does not include limits and is related to <u>atomic</u> operations on <u>different elements</u> rather than commutative operations.

With regard to Claim 7 the Examiner also states that "Gostanian discloses a method in which it is determined that the result of two operations is listed as commutative and thus are allowed to execute if it is determined that the result would be identical if the operations were run in any order (column 10, line 15-column 11, line 10 of Gostanian)." The Applicant traverses this statement. In the cited text <u>Gostanian</u> *defines* commutative operations as those wherein "the final state of the replicated database server does not depend on the order in which this transaction executes relative to all other transactions." This is a definition of commutativity not a method of determining which transitions are commutative. As a definition, it could only be used to determine the commutativity of a transaction by trial and error. However, in <u>Gostanian</u>, the classification of a transaction as being commutative is <u>not</u> dependent on performing transactions in different orders and comparing results. Rather a transaction is classified as commutative, or non-commutative, based on transaction types, i.e., whether transactions include addition, multiplication or some other type. Then based on this classification, it is assumed that the final state is independent or dependent on execution order. This is illustrated in the examples given on Col. 10, lines 40 – 67 of Gostanian.

As discussed above with respect to Claim 12, the Applicant is unable to find any teaching within the cited art wherein two or more operations are categorized as commutative or non-commutative responsive to whether either violate a limit when executed. Specifically, the Applicant is unable to find any teaching of "*categorizing the two or more computational operations determined to violate the limit as not commutative*" as recited in Claim 7. The Applicant therefore requests that the Examiner more particularly point out teaching of this and the other limitations of Claim 7, or allow Claim 7.

Regarding Claim 8.

Claim 8 includes limitations similar to those recited in Claim 7. The Applicant, therefore, believes that Claim 8 is allowable for the same reasons discussed above with respect to Claim 7.

Regarding New Claim 14.

Claim 14 is supported by the same parts of the application as filed as are Claims 7 and 12 and is believed to be allowable for at least the same reasons as Claims 7 and 12.

The Applicant believes that all pending claims are allowable and respectfully request that the Examiner issue a Notice of Allowance. Should the Examiner have questions, the Applicant's undersigned representative may be reached at the number provided below.

Respectfully submitted,
Clifford L. Hersh

[illegible]

Date: March 17, 2005

Steven M. Colby, Reg. No. 50,250
Carr & Ferrell *LLP*
2200 Geng Rd.
Palo Alto, CA 94303
Phone (650) 812-3424
Fax (650) 812-3444